Learning DevOps

Overview About CI - CD - CD (Continuous Integration, Continuous deployment, Continuous Delivery)

What is CI

Code 
Unit 
Commit 
Tests 
Code 
Quality 
Package 
Integration 
Tests 

Objective of CI is to detect the issues earlier than later. This is achieved by continuous feedback we get from the above steps.

What is CD (Continuous deployment)

Commit 
Unit 
Tests 
Integration 
Package 
Tests 
Deploy 
Automated 
Tests 

Compared to CI here in CD we have take additional steps by means of deploy and automated tests.

Deploy into Dev/QA environment and in addition we run few more automated tests (Smoke tests)

What is CD continuous delivery:

Unit 
Testing 
Approval 
Deploy 

CD (continuous delivery takes this process a step further)

In Continuous delivery, along with packaging and deploying and running our smoke tests, we'd setup a pipeline in such a way that on the appropriate approvals on that specific environment, we’ll go ahead and deploy it out to the staging environment.

Lets assume in the above pic if the white circle is the UAT environment, on proper approval of tests from UAT, we go ahead and deploy it out in the staging environment/ production.

By following this approach in continuous delivery we will be able to deploy the code to the production only with few approvals.

As soon as the code is committed to GIT (repository), all the steps would be run and if the QA team and UAT team is ready, and they'd do the manual approvals, and the code would be immediately deployed on to stage and production environments.

Popular tools for achieving CI and CD are Jenkins and Azure DevOps.

The means to implement these things using Azure DevOps is via pipelines.

Each of the above steps / process shown here is a step in a pipeline.

We can create Continuous integration pipelines/ continuous delivery / continuous deployment pipelines.

Various tools used during CI pipelines:

Code Commit -----> Git hub / some other private repository.

Unit tests -----> Mocha/ Jasminw(JavaScript), C++(SonarQube/ Line/ other unit test frameworks), Python (pytest), Java (Junit)

Integration tests -----> Cucumber, Selenium, protractor (Writing automated tests for few modules integrated together) (BDD is another example)

Package -----> Creating deployable unit for our application, tools used are npm (JavaScript), PIP (python), Maven/ Gradle (Java)

Once we have the application package ready we can easily deploy it to any of our environments either using Jenkins or Azure DevOps.

Once we deploy the code to our environment, we can run additional automated tests, these automated tests could be (smoke test/ Load test/ performance test)

We can see in soon how we can implement manual approvals to deploy code into a specific environment.

These approvals will be based on if the automated tests runs fine and when manual tests are also fine the approval can go through and the code will be deployed automatically to the next environment.

For Ex: If the code current approval is in QA stage then after successful approval the code will be deployed to UAT environment or staging environment.

If the approval is from UAT stage then on successful approval, the code will be deployed to production.

CICD pipeline also helps in automating the deployment of the code packages to different environments.

Azure DevOps Amazon EKS with Terraform

In this section we are going to see how to setup a K8s Cluster in AWS using Terraform.

The cluster will be setup using AWS EKS (Elastic container service for Kubernetes).

We will also see how to deploy microservice inside the cluster.

We need 2 pipelines for this purpose.

1 for provisioning the cluster and the other one for the CI/CD (Build the docker image, and deploy it K8s cluster).

These pipelines are going to be running in Azure DevOps.